FIG. 1

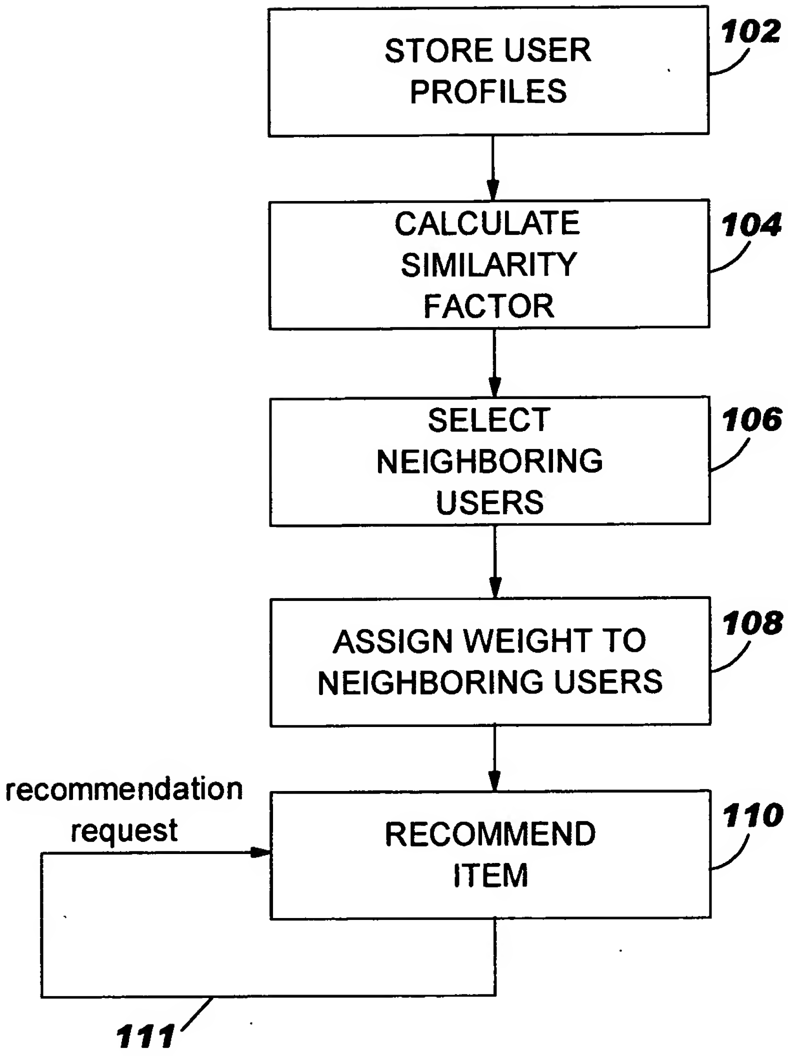
STORE USER PROFILES
102
CALCULATE SIMILARITY FACTOR
104
SELECT NEIGHBORING USERS
106
ASSIGN WEIGHT TO NEIGHBORING USERS
108
recommendation request
RECOMMEND ITEM
110
111

FIG. 2

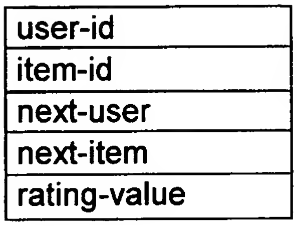
user-id
item-id
next-user
next-item
rating-value

FIG. 3

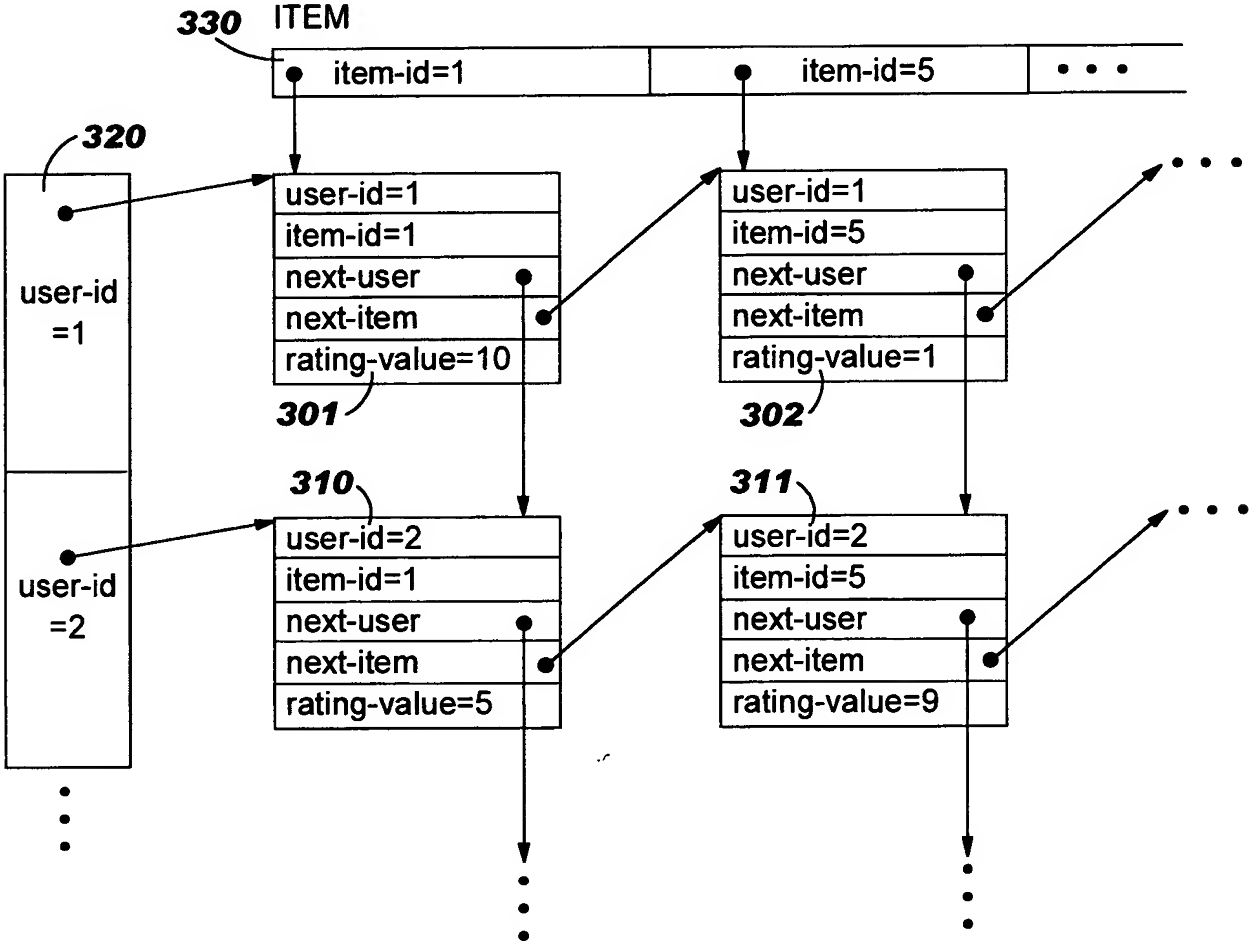
330
ITEM
item-id=1
item-id=5
• • •
320
user-id =1
user-id =2
user-id=1
item-id=1
next-user
next-item
rating-value=10
301
user-id=1
item-id=5
next-user
next-item
rating-value=1
302
310
user-id=2
item-id=1
next-user
next-item
rating-value=5
311
user-id=2
item-id=5
next-user
next-item
rating-value=9

FIG. 4

```
calc(u,selected)
define used[v]=false for each user v
used[u]=true
initialize list N as empty list

foreach rating ru of list USER[u] do
  if (selected[ru.item-id])
  foreach rating ri of list ITEM[ru.item-id] do
        u'=ri.user-id
        if (used[u']==false
             used[u']=true
             append tuple (u',similarity)u,u',selected)) to list N

sort N by value t.s of each tuple t=(t.u,t.s) by quicksort/heapsort

return N


similarity(x,y,selected)
this function is returning the similarity between user x and user y,
   eg. the Pierce coefficient.
It will be computed only on the items "it" for which selected[it]==true
```

FIG. 5

```
501.calc(u,selected)
502.if there is no computed list N(u) for user u just do the "normal" calc(u,selected)
503.
504.if timestamp(last update u) > timestampe(N(u)) do the "normal" calc(u,selected)
505.
506.foreach tuple t=(t.u,t.s) of N(u) do
507.        if (timestamp(last update t.u) > timestamp(N(u))
508.            t.s = similarity(u,t.u,selected)
509.
510.sort N(u) by bubble sort if the number of updates is small,
    otherwise by quicksort/heapsort
511.update timestamp of N(u)
512.return N(u)
```